МОСКОВСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ  
им. Н.Э. Баумана

Факультет “Информатика и системы управления”  
Кафедра “Системы обработки информации и управления”

A logo with a gold star and blue ribbon

Description automatically generated

Дисциплина “Парадигмы и конструкции языков программирования”

Отчет по рубежному контролю №1  
Вариант Г13

**Выполнил:**Студент группы ИУ5-35Б  
Ким А. А.  
**Проверил:**Гапанюк Ю. Е.

подпись, дата

\_\_\_\_\_\_\_\_/\_\_\_\_\_\_\_\_\_

Москва 2025

**Задание**

В рамках рубежного контроля №1 по курсу «Парадигмы и конструкции языков программирования» требовалось разработать программу на языке Python.

**Цель работы:**

1. Создать два класса данных, соответствующие предметной области **«Книга» (Класс 1) и «Библиотека» (Класс 2)** (Вариант №13).

2. Реализовать связь «один-ко-многим» между классами.

3. Создать дополнительный класс для реализации связи «многие-ко-многим».

4. Создать списки объектов классов, содержащие тестовые данные (3-5 записей).

5. Реализовать три запроса в соответствии с **Вариантом Г**, адаптировав их под выбранную предметную область.

6. При реализации запросов использовать функциональные возможности Python (list/dict comprehensions, функции высших порядков).

**Запросы:**

1. «Книга» и «Библиотека» связаны соотношением «один-ко-многим». Вывести список всех библиотек, у которых название начинается с буквы «А», и список книг, находящихся в них.

2. «Книга» и «Библиотека» связаны соотношением «один-ко-многим». Вывести список библиотек с максимальным количеством страниц книг в каждой библиотеке, отсортированный по максимальному количеству страниц.

3. «Книга» и «Библиотека» связаны соотношением «многие-ко-многим». Вывести список всех связанных книг и библиотек, отсортированный по библиотекам.

**Листинг программы**

*# Вариант Г 13*from operator import itemgetter  
  
class Book:  
 def \_\_init\_\_(self, id, title, pages, library\_id):  
 self.id = id  
 self.title = title  
 self.pages = pages  
 self.library\_id = library\_id  
  
class Library:  
 def \_\_init\_\_(self, id, name):  
 self.id = id  
 self.name = name  
  
class BookLibrary:  
 def \_\_init\_\_(self, library\_id, book\_id):  
 self.library\_id = library\_id  
 self.book\_id = book\_id  
  
*# Библиотеки*libraries = [  
 Library(1, 'Центральная библиотека'),  
 Library(2, 'Академическая библиотека'),  
 Library(3, 'Детская библиотека'),  
 Library(11, 'Арт-библиотека'),  
 Library(22, 'Абонемент научной литературы'),  
 Library(33, 'Английская библиотека'),  
]  
  
*# Книги*books = [  
 Book(1, 'Война и мир', 1225, 1),  
 Book(2, 'Преступление и наказание', 671, 2),  
 Book(3, 'Мастер и Маргарита', 480, 3),  
 Book(4, '1984', 328, 3),  
 Book(5, 'Гарри Поттер', 500, 3),  
]  
  
*# Связи многие-ко-многим*books\_libraries = [  
 BookLibrary(1, 1),  
 BookLibrary(2, 2),  
 BookLibrary(3, 3),  
 BookLibrary(3, 4),  
 BookLibrary(3, 5),  
 BookLibrary(11, 1),  
 BookLibrary(22, 2),  
 BookLibrary(33, 3),  
 BookLibrary(33, 4),  
 BookLibrary(33, 5),  
]  
  
  
def main():  
 *# Соединение данных один-ко-многим* one\_to\_many = [(b.title, b.pages, lib.name)  
 for lib in libraries  
 for b in books  
 if b.library\_id == lib.id]  
  
 *# Соединение данных многие-ко-многим* many\_to\_many\_temp = [(lib.name, bl.library\_id, bl.book\_id)  
 for lib in libraries  
 for bl in books\_libraries  
 if lib.id == bl.library\_id]  
  
 many\_to\_many = [(b.title, b.pages, lib\_name)  
 for lib\_name, lib\_id, book\_id in many\_to\_many\_temp  
 for b in books if b.id == book\_id]  
  
 print('Задание Г1')  
 print('Список всех библиотек, у которых название начинается с буквы "А", и список книг в них:')  
  
 libs\_with\_a = list(filter(lambda i: i[2].startswith('А'), many\_to\_many))  
  
 *# Группируем по библиотекам* res\_1 = {}  
 for book\_title, pages, lib\_name in libs\_with\_a:  
 if lib\_name not in res\_1:  
 res\_1[lib\_name] = []  
  
 if book\_title not in res\_1[lib\_name]:  
 res\_1[lib\_name].append(book\_title)  
  
 for lib, books\_list in sorted(res\_1.items()):  
 print(f'{lib}: {books\_list}')  
  
 print('\nЗадание Г2')  
 print('Список библиотек с максимальным количеством страниц книг в каждой библиотеке, отсортированный по максимальному количеству страниц:')  
  
 res\_2\_unsorted = []  
  
 *# Перебираем все библиотеки* for lib in libraries:  
  
 lib\_books = list(filter(lambda i: i[2] == lib.name, many\_to\_many))  
  
  
 if len(lib\_books) > 0:  
 lib\_pages = [pages for \_, pages, \_ in lib\_books]  
 lib\_pages\_max = max(lib\_pages)  
 res\_2\_unsorted.append((lib.name, lib\_pages\_max))  
 else:  
 res\_2\_unsorted.append((lib.name, 0))  
  
 *# Сортировка по максимальному количеству страниц (по убыванию)* res\_2 = sorted(res\_2\_unsorted, key=itemgetter(1), reverse=True)  
  
 for lib, max\_pages in res\_2:  
 print(f'{lib}: {max\_pages} стр.')  
  
 print('\nЗадание Г3')  
 print('Список всех связанных книг и библиотек, отсортированный по библиотекам:')  
  
 *# Сортируем по названию библиотеки* res\_3 = sorted(many\_to\_many, key=itemgetter(2))  
  
 grouped\_result = {}  
 for book\_title, pages, lib\_name in res\_3:  
 if lib\_name not in grouped\_result:  
 grouped\_result[lib\_name] = []  
 grouped\_result[lib\_name].append(f"{book\_title} ({pages} стр.)")  
  
 for lib\_name, books\_list in sorted(grouped\_result.items()):  
 print(f'{lib\_name}: {books\_list}')  
  
  
if \_\_name\_\_ == '\_\_main\_\_':  
 main()

**Скриншоты работы программы**

